Scheduling divisible workloads on heterogeneous platforms under Bounded Multi-Port model

Olivier Beaumont, Nicolas Bonichon, Lionel Eyraud-Dubois

LAboratoire Bordelais de Recherche en Informatique
Equipe CEPAGE (INRIA)

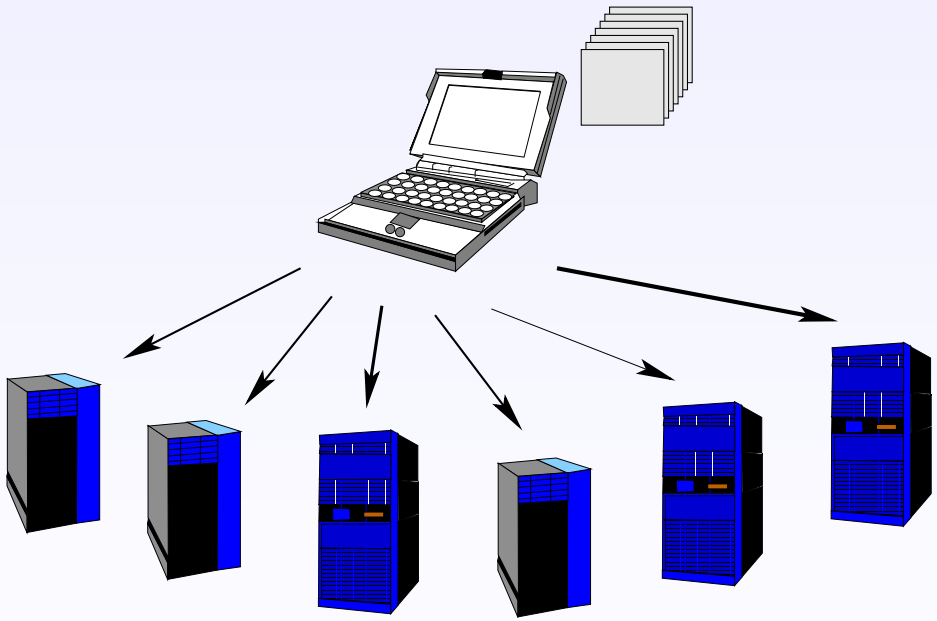
Alpage
January 2007

Outline

Outline

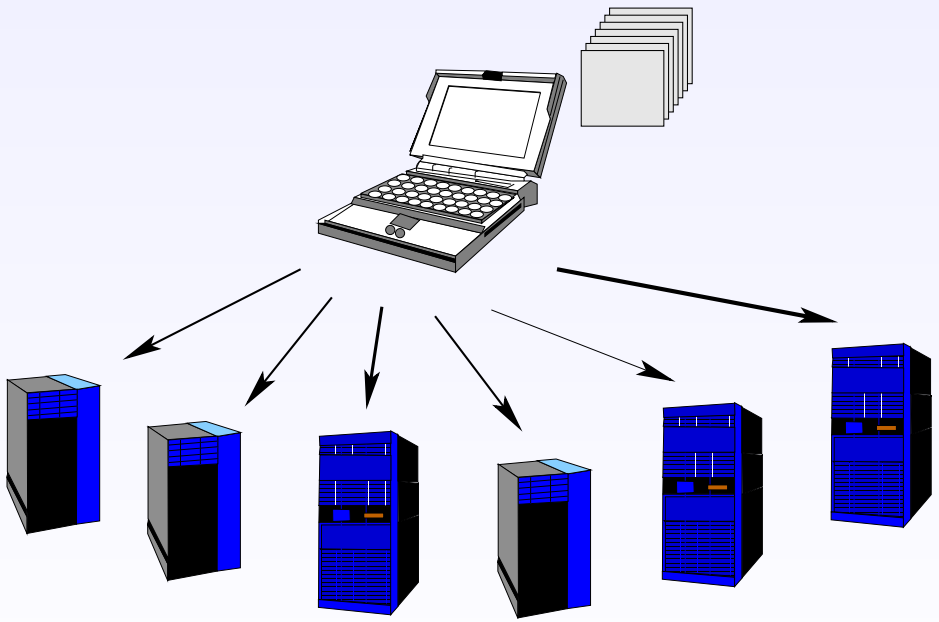
Introduction

- One master, holding a large number of identical tasks
- Some workers



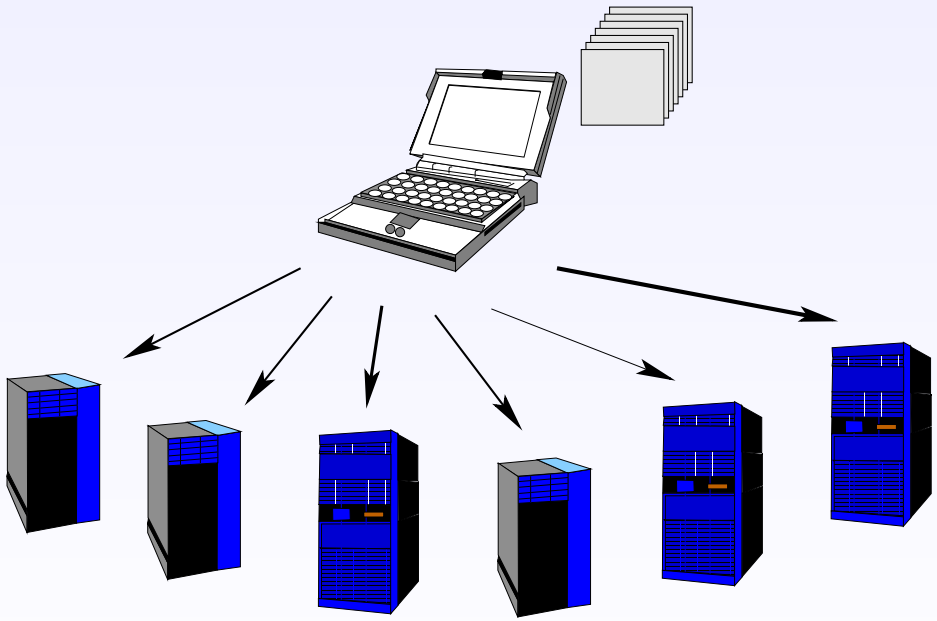
Introduction

- One master, holding a large number of identical tasks
- Some workers
- Heterogeneity in computing speed and bandwidth



Introduction

- One master, holding a large number of identical tasks
- Some workers
- Heterogeneity in computing speed and bandwidth
- Distribute work to workers



Assumption: divisible load

- Important relaxation of the problem

Assumption: divisible load

- Important relaxation of the problem
- Master holding N tasks

Assumption: divisible load

- Important relaxation of the problem
- Master holding N tasks
- Worker P_i will get a fraction $\alpha_i \times N$ of these tasks

Assumption: divisible load

- Important relaxation of the problem
- Master holding N tasks
- Worker P_i will get a fraction $\alpha_i \times N$ of these tasks
- α_i is **rational**, tasks are divisible

Assumption: divisible load

- Important relaxation of the problem
- Master holding N tasks
- Worker P_i will get a fraction $\alpha_i \times N$ of these tasks
- α_i is **rational**, tasks are divisible
- $\Rightarrow$ possible to derive analytical solutions (tractability)

Assumption: divisible load

- Important relaxation of the problem
- Master holding N tasks
- Worker P_i will get a fraction $\alpha_i \times N$ of these tasks
- α_i is **rational**, tasks are divisible
- $\Rightarrow$ possible to derive analytical solutions (tractability)
- In practice, reasonable assumption with a large number of tasks

Assumption: divisible load

- Important relaxation of the problem
- Master holding N tasks
- Worker P_i will get a fraction $\alpha_i \times N$ of these tasks
- α_i is **rational**, tasks are divisible
- $\Rightarrow$ possible to derive analytical solutions (tractability)
- In practice, reasonable assumption with a large number of tasks
- Worker P_i can start processing tasks only once it has received the whole data (1-round)

Background on Divisible Load Scheduling: 1-port

- Linear cost model: X units of work:
  - sent to P_i in $X \times c_i$ time units
  - computed by P_i in $X \times w_i$ time units

Background on Divisible Load Scheduling: 1-port

- Linear cost model: X units of work:
  - sent to P_i in $X \times c_i$ time units
  - computed by P_i in $X \times w_i$ time units

Results:

- Bus network $\Rightarrow$ all processors work and finish at the same time
  order does not matter closed-formula for the makespan (Bataineh, Hsiung & Robertazzi, 1994)
- Result extended for homogeneous tree: a subtree reduces to one single worker
- Heterogeneous star network:
  all processors work and finish at the same time
  order matters:
  - largest bandwidth first (whatever the computing power)

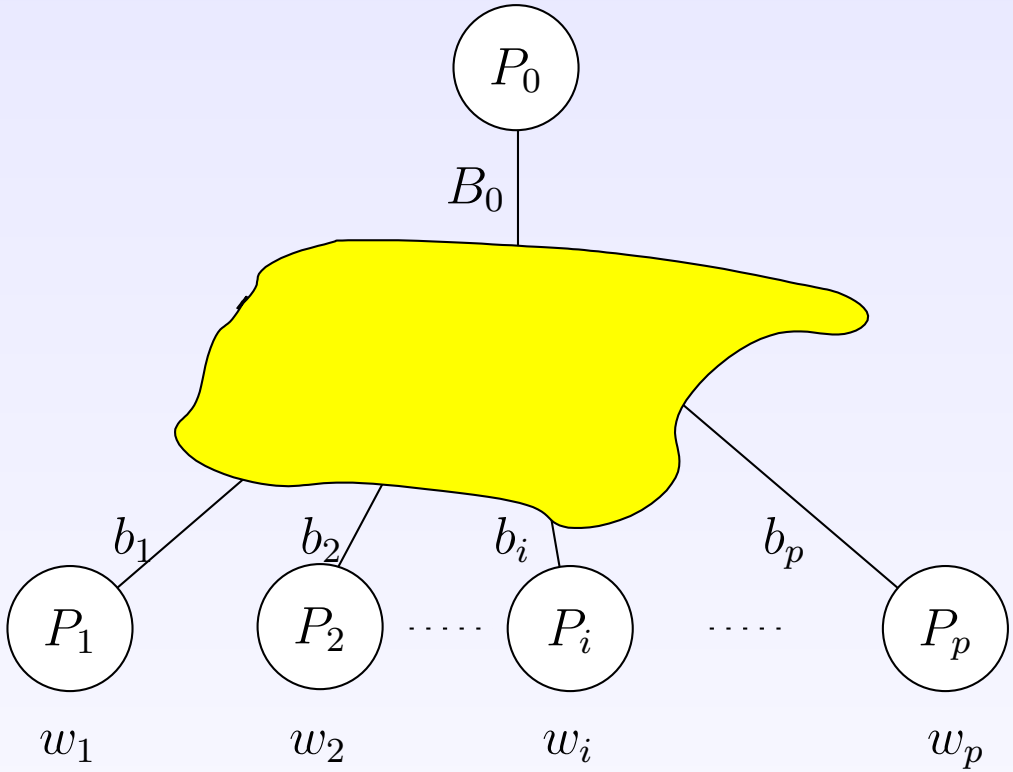
Background on Divisible Load Scheduling: 1-port

- With an affine cost model: X units of work:
  - sent to P_i in $C_i + X \times c_i$ time units
  - computed by P_i in $W_i + X \times w_i$ time units

  $\Rightarrow$ not all processors participate to the computation
  selecting the resources is hard:
  - computing the optimal schedule on a star with affine cost model is NP-hard (Casanova, Drodowski, Legrand & Yang, 2005)

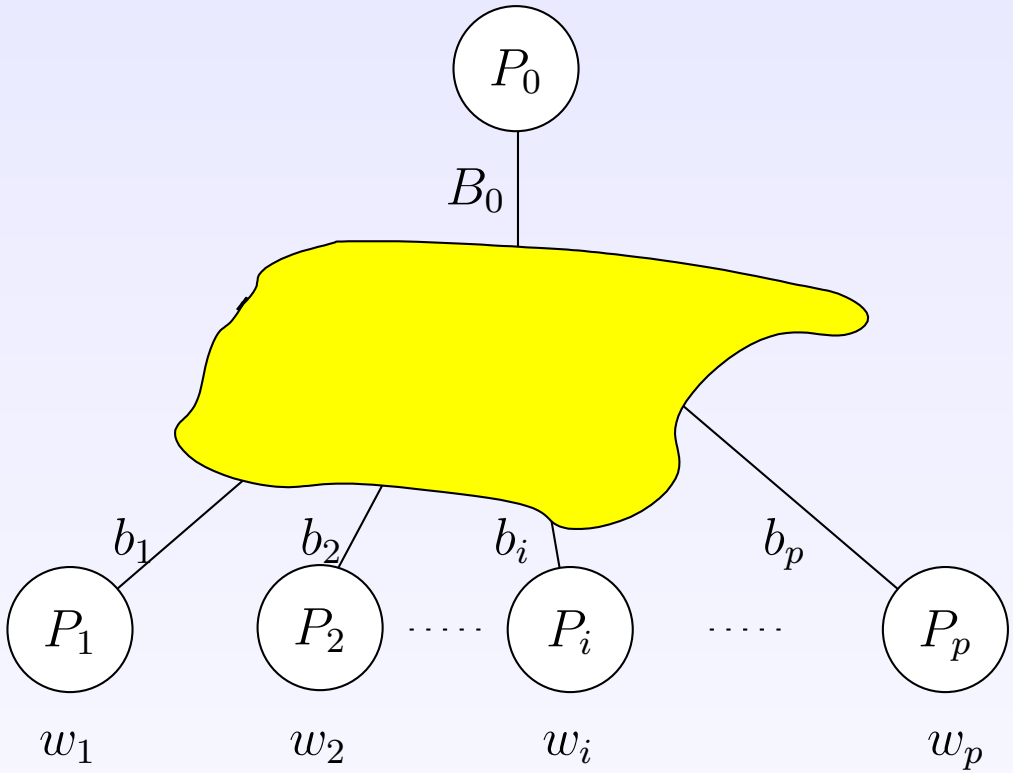
Outline

Platform



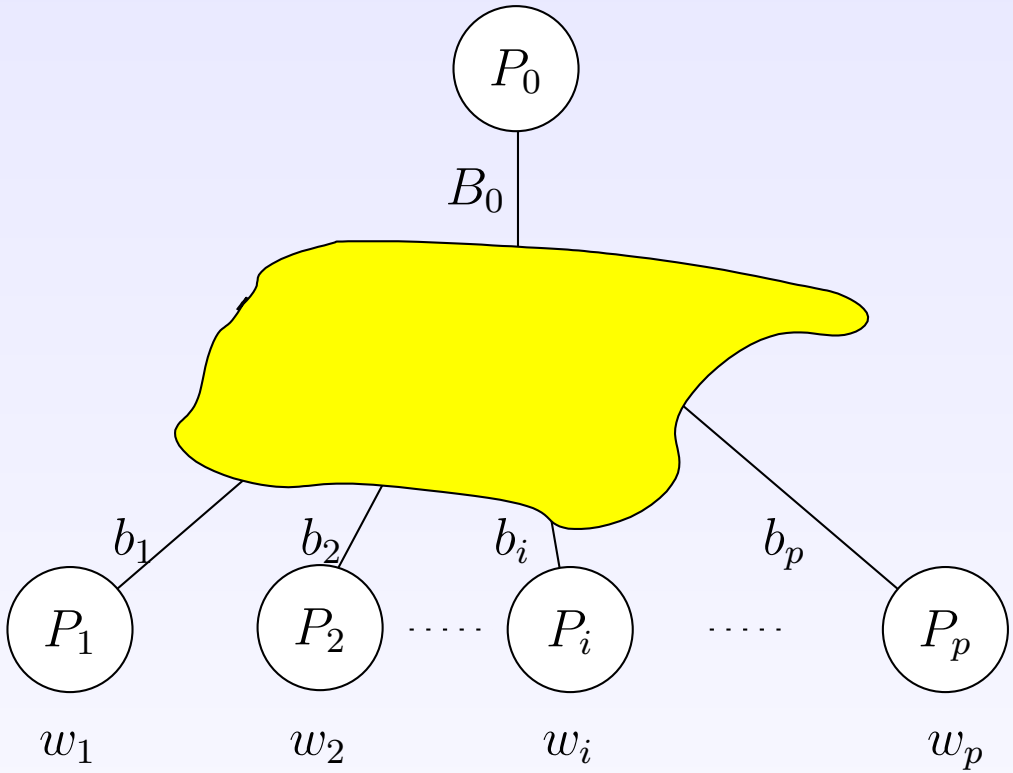
- B_0: output bandiwth of the master processor.

Platform



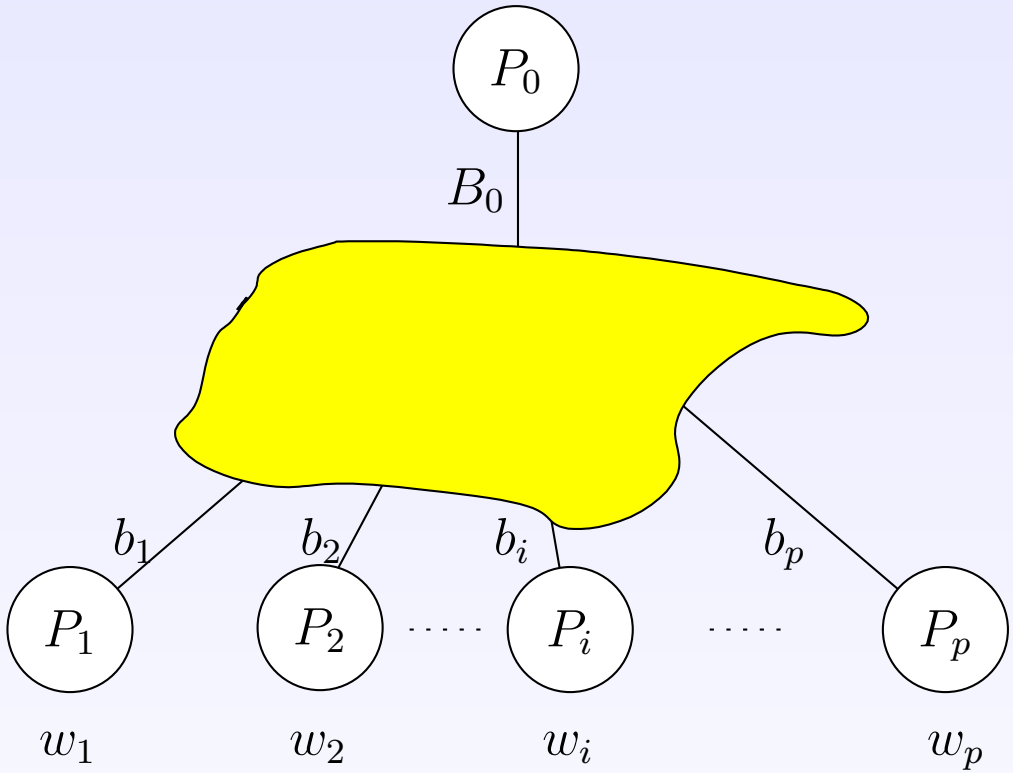
- B_0: output bandiwth of the master processor.
- b_i: intput bandwith of the slave P_i.

Platform



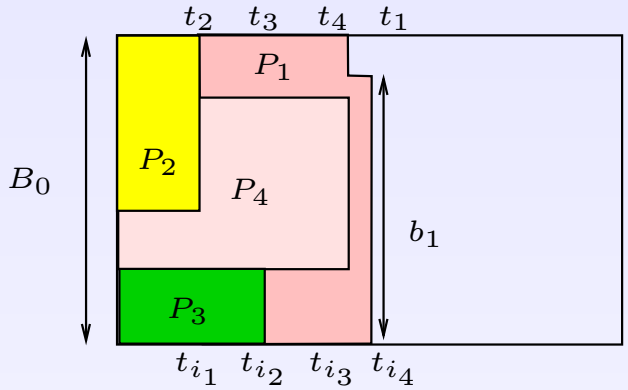
- B_0: output bandiwth of the master processor.
- b_i: intput bandwith of the slave P_i.
- w_i: time needed by P_i to compte a unit-task.

Platform



- B_0: output bandiwth of the master processor.
- b_i: intput bandwith of the slave P_i.
- w_i: time needed by P_i to compte a unit-task.
- Assumption: No interaction between communications.

Bounded Multi-Port Model



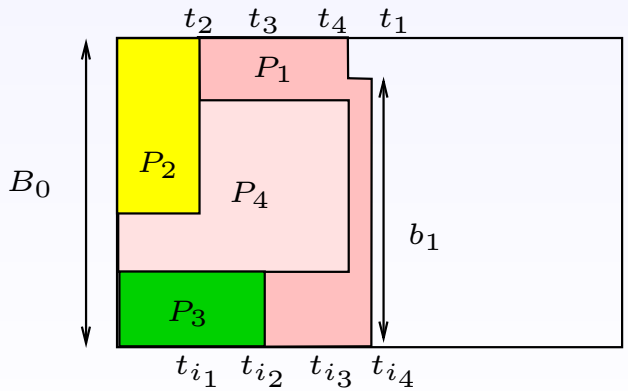
- Notations:
  - $b'_i(t)$: actual bandwidth used at time t by the communication between P_0 and P_i.
  - t_i: the time when processor P_i stops communicating.
  - q_i: the fractionnal number of tasks processed by P_i: $q_i = \int_0^1 \sum_i b'_i(t)dt$.
- Constraints:
  - input bandwidth: $\forall t, b'_i(t) \leqslant b_i$.
  - output bandwidth: $\forall t, \sum_i b'_i(t) \leqslant B_0$.
  - processing time: $t_i + w_i.q_i \leqslant 1$
- Goal: Maximizing $\sum_i q_i$

Normal Form

Definition

A schedule is said to be in *normal form* if

1. all processors are involved in the processing of tasks,
2. all slaves start processing tasks immediately after the end of the communication with the master (at time t_i) and stop processing at time 1,
3. during each time slot $]t_{i_k}, t_{i_{k+1}}]$ the bandwidth used by any processor is constant.

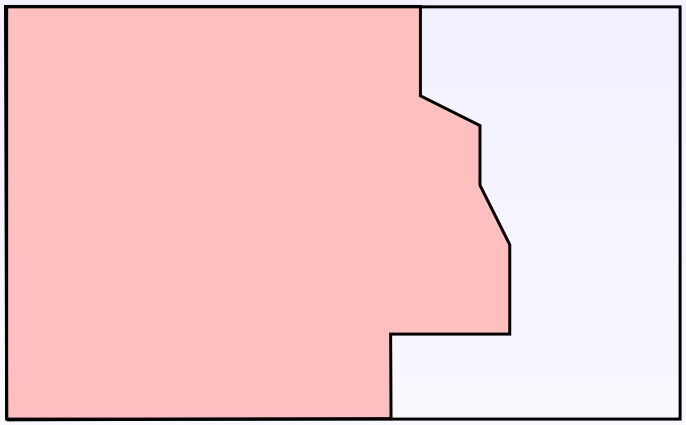


Lemma 1

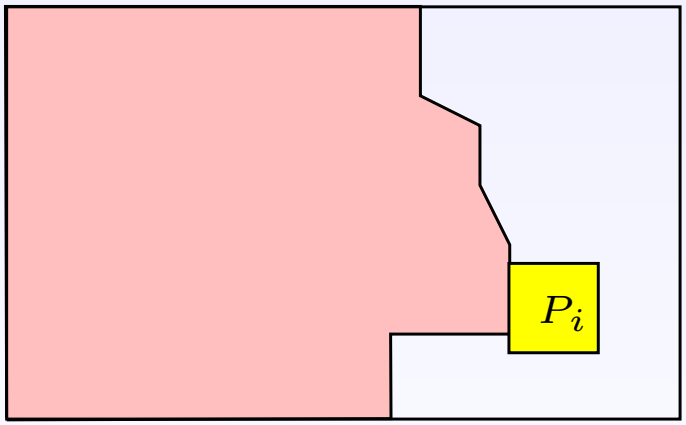
Lemma

In an optimal schedule, all processors take part in the computations.

Proof:



Original schedule



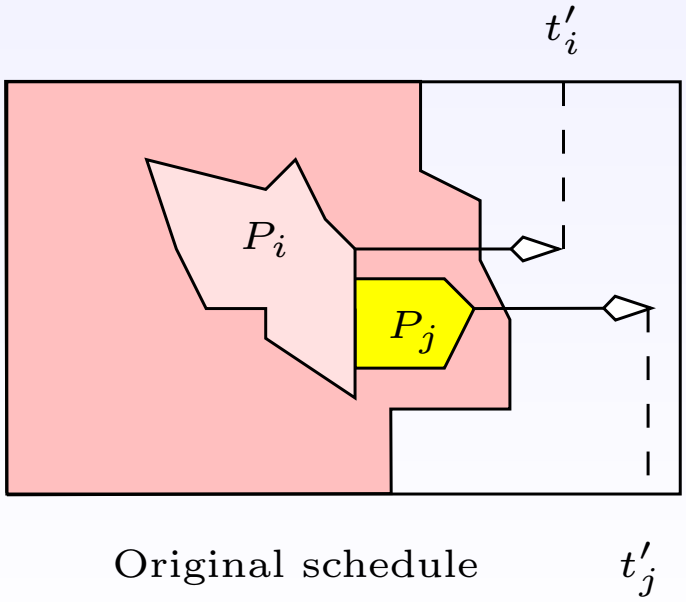
New schedule

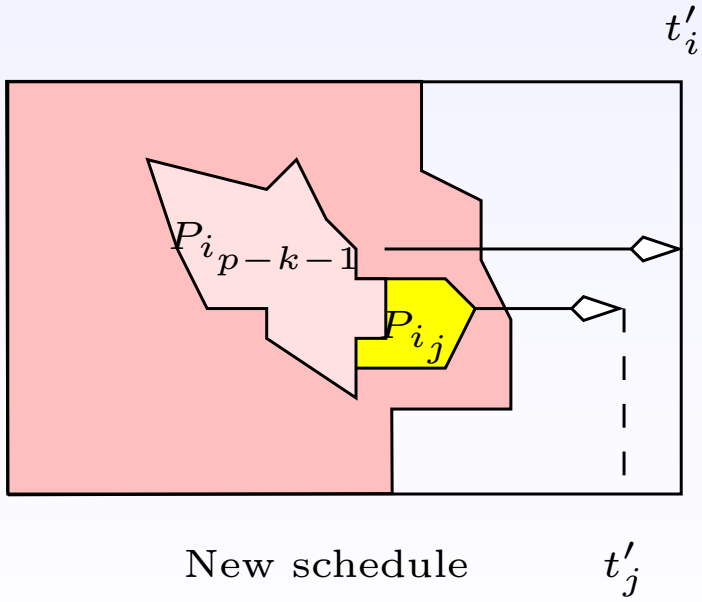
Lemma 2

Lemma

In an optimal schedule, slaves start processing tasks immediately after the end of the communication with the master and stop processing at time 1, i.e. there is no idle time between the end of the communication and the deadline.

Proof: By induction on the last slave that stop processing before $t = 1$.



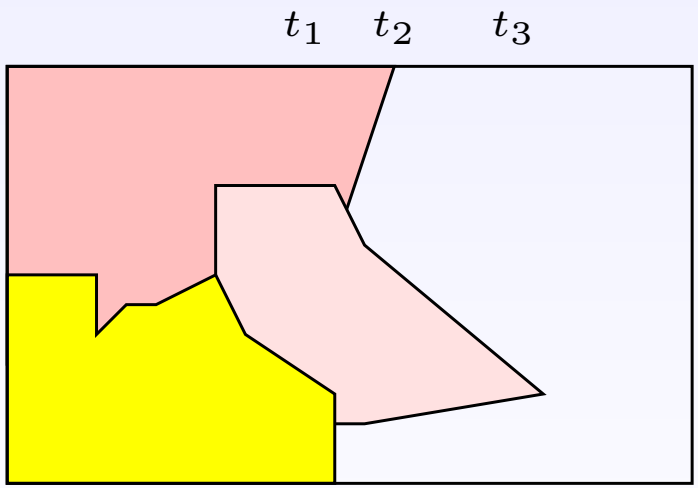


Lemma 3

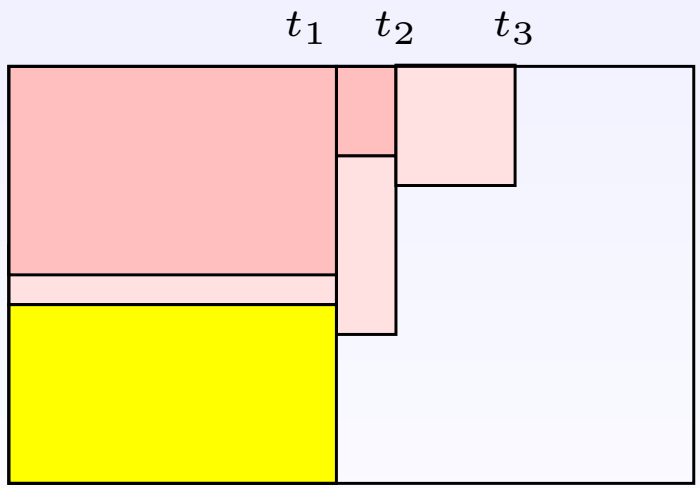
Lemma

There exists an optimal schedule such that during each time slot $]t_{i_k}, t_{i_{k+1}}]$ *the bandwidth used by any processor is constant.*

Proof:



Original schedule



New schedule

Normal Form

Theorem

We can restrict the search of optimal solutions within the valid schedules in normal form.

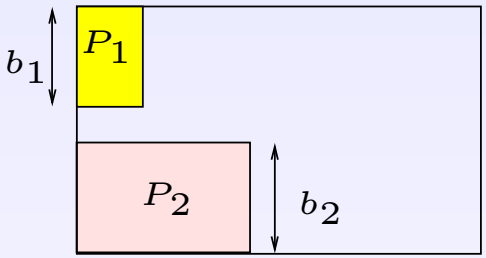
Proof.

The proof of the theorem comes directly from Lemmas 1, 2 and 3. □

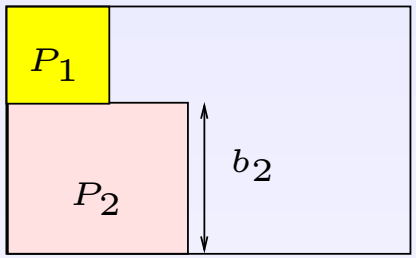
Outline

Different cases with 2 slaves

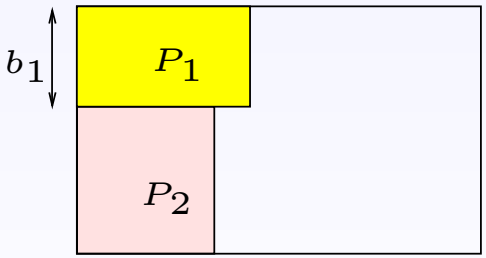
$b_2 < B_0 \frac{w_1}{w_1+w_2}$ and $b_1 < B_0 \frac{w_2}{w_1+w_2}$



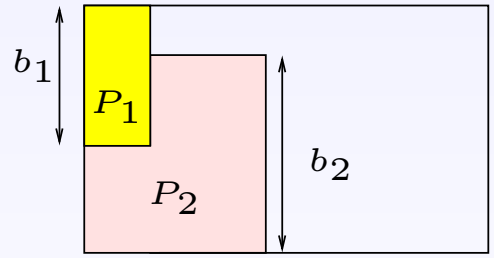
$b_2 < B_0 \frac{w_1}{w_1+w_2}$ and $b_1 \geqslant B_0 \frac{w_2}{w_1+w_2}$

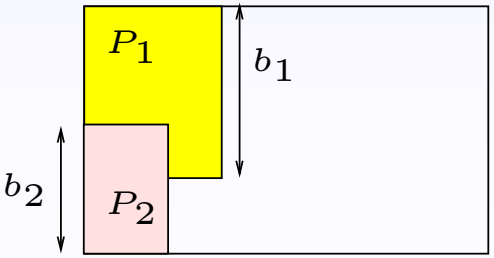


$b_2 \geqslant B_0 \frac{w_1}{w_1+w_2}$ and $b_1 < B_0 \frac{w_2}{w_1+w_2}$

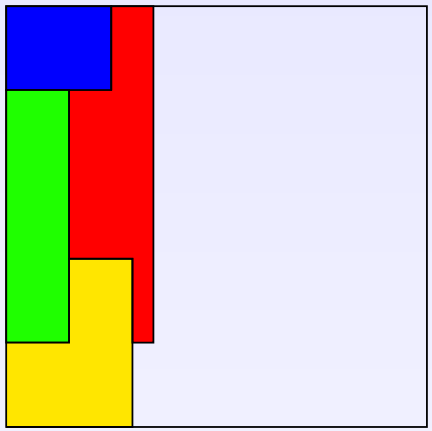


$b_2 \geqslant B_0 \frac{w_1}{w_1+w_2}$ and $b_1 \geqslant B_0 \frac{w_2}{w_1+w_2}$

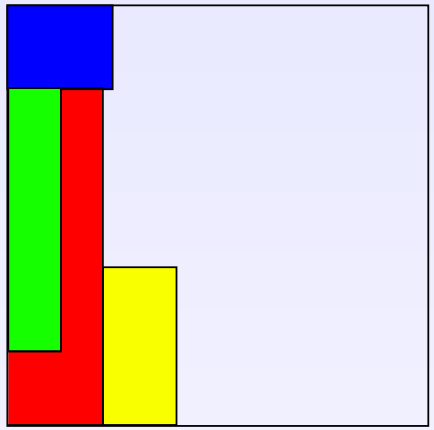




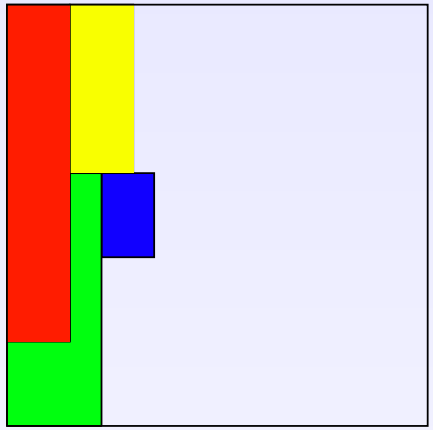
Extension of 1-port solutions ?



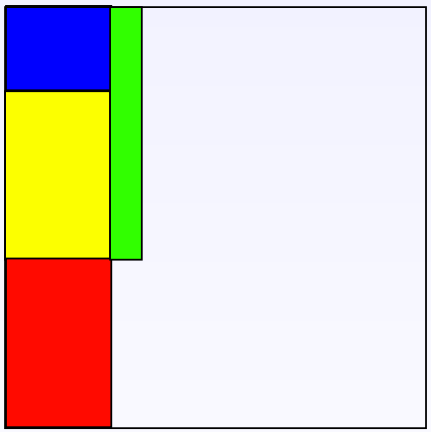
1324 (decreasing w): 2.067



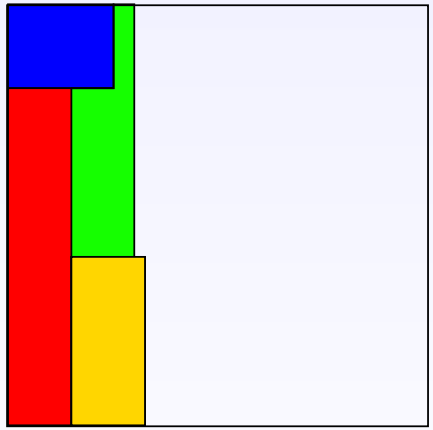
1342 (decreasing w): 2.03



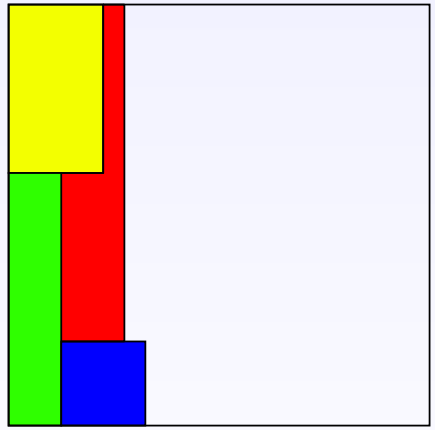
4321 (decreasing b): 2.042



1243 (increasing $b * w$): 2.03



1423 (best solution): 2.078



2314 (best solution): 2.078

$B_0 = 5$

- $1 : b = 1, w = 2$
- $2 : b = 2, w = 1$
- $3 : b = 3, w = 1.5$
- $4 : b = 4, w = 1$

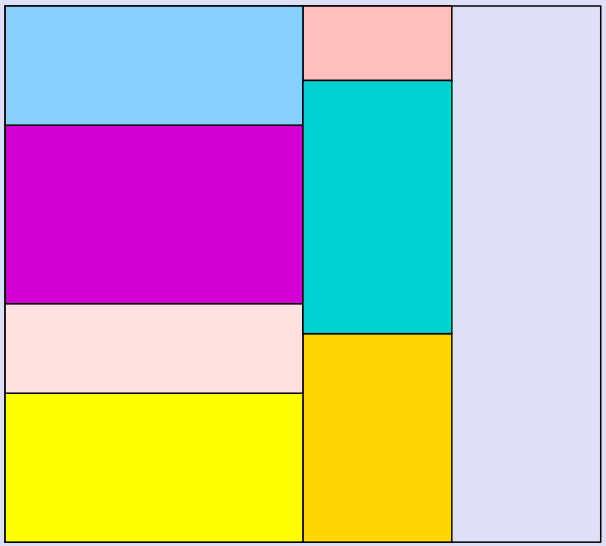
Outline

Theorem

BoundedMPDivisible *is NP-complete.*

Proof.

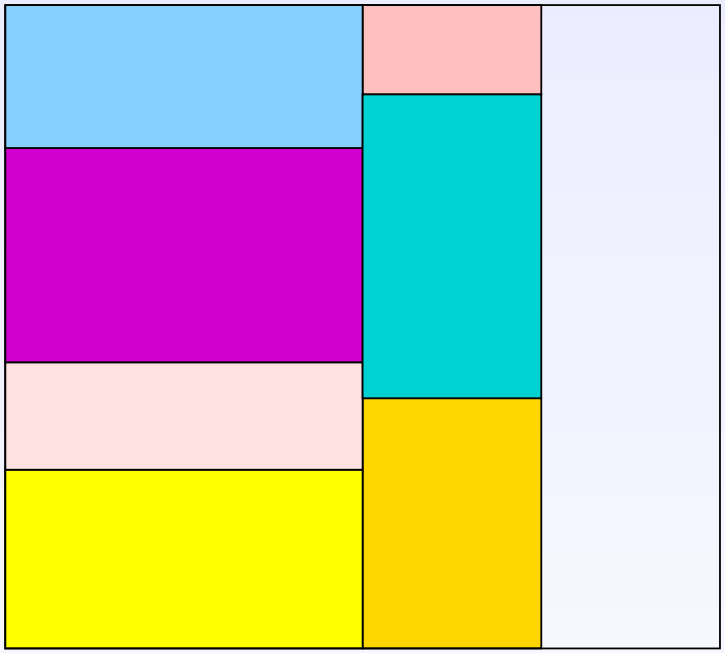
Reduction from 2-Partition. We build an instance of BoundedMPDivisible with $w_i = 1/b_i$ and $\sum_i b_i = 2B_0$.



$Q = 3B_0/4$

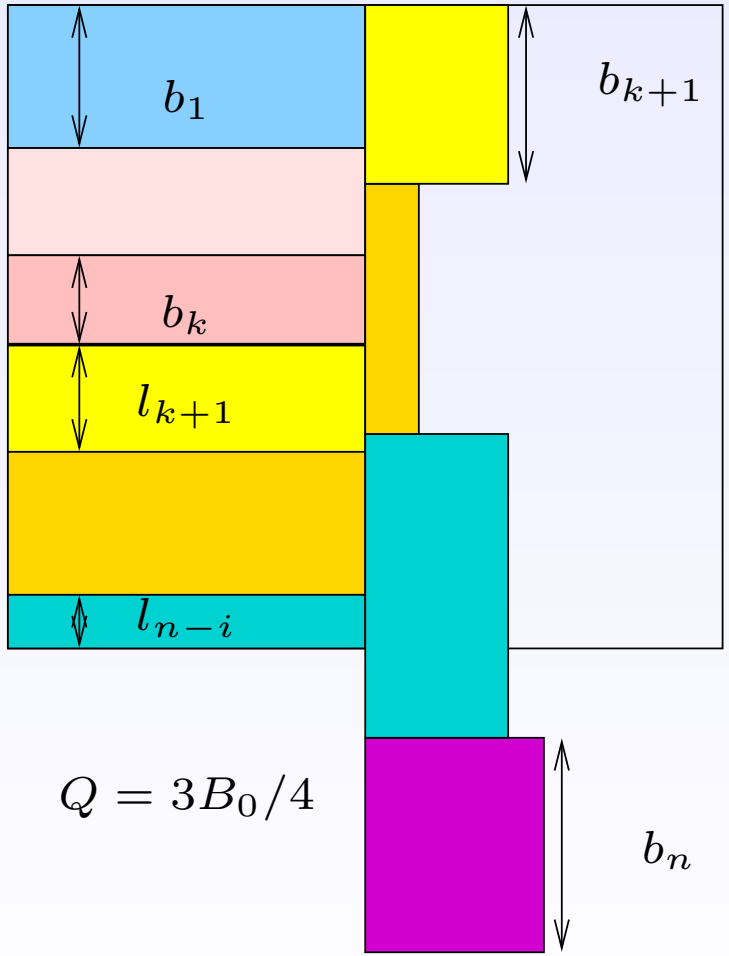
NP-Completeness: proof

(a) Optimal solution



$Q = 3B_0/4$

(b) Hypothetic solution



$Q = 3B_0/4$

Outline

Conclusion

- A more realistic model for Divisible Load Scheduling
- Exhaustive study with two processors $\Rightarrow$: no intuition :-(
- NP-Completeness
- Future work:
  - Approximation algortihm
  - Experiments